Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Claims 1-20 (canceled).

Claim 21 (currently amended): A method for generating a tuple representing relationships among ~~nodes~~ connectors in a network, the method comprising:

identifying a link directly coupling a host to a first port of a ~~node~~first connector;

identifying an intermediate connection which indirectly couples the host to an intermediate ~~node~~port of an intermediate connector, wherein the intermediate ~~node~~ connector is not aware ~~that the node exists~~of a connection to the first connector and the first connector is not aware of a connection to the intermediate connector; and

generating a new tuple identifying a relationship between the ~~node~~ first connector and the intermediate ~~node~~ connector based on the identified intermediate connection and the direct link, wherein the new tuple indicates that the ~~node~~ first connector is directly coupled to the intermediate ~~node~~connector.

Claim 22 (currently amended): The method of claim 21, wherein the host is a singly-heard-host which is the only host heard on the first port of the ~~node~~first connector.

Claim 23 (previously presented): The method of claim 22, wherein the singly-heard-host is at least one of a workstation, a personal computer, a terminal and a printer.

Claim 24 (previously presented): The method of claim 21, wherein the tuple and the new tuple contain data associated with a topology of the network.

Claim 25 (currently amended): The method of claim 21, wherein generating the new tuple comprises:

determining that the ~~node~~ first connector is directly coupled to the intermediate ~~node~~ connector via a second port of the ~~node~~first connector.

Claim 26 (currently amended): The method of claim 25, further comprising:

storing the new tuple in the ~~the~~ intermediate ~~node~~connector.

Claim 27 (currently amended): The method of claim 25, further comprising:

storing the new tuple in the ~~the node~~first connector.

Claim 28 (currently amended): The method of claim 21, further comprising:

determining whether the host is heard only by the first port of the ~~node~~first connector;

if the host is heard only by the first port of the ~~node~~first connector, classifying the new tuple as a singly-heard host link tuple.

Claim 29 (currently amended): The method of claim 28, further comprising:

determining if another ~~node~~ connector hears the host as a singly-heard host; and

if another ~~node~~ connector hears the host, classifying the new tuple as a singly-heard conflict link tuple; and

resolving a conflict associated with the host between the ~~node~~ first connector and the another ~~node~~connector.

Claim 30 (currently amended): The method of claim 28, ~~futher~~ further comprising:

generating an extra host link tuple for the intermediate ~~node~~ connector indirectly coupled to the host via the intermediate connection.

Claim 31 (currently amended): The method of claim 30, further comprising:

examining the singly heard host link tuple and the extra host link tuple; and

based on the examining if the ~~node~~ first connector is determined to be connected to the host and the intermediate ~~node~~ connector is determined to be connected to the host, generating a conn-to-conn link tuple between the ~~node~~ first connector and the intermediate ~~node~~connector.

Claim 32 (currently amended): A system for generating a tuple representing relationships among ~~nodes~~ connectors in a network, the system comprising:

a first ~~node~~ connector directly coupled to a host via a first port;

an intermediate ~~node~~ connector indirectly coupled to the host via an intermediate connection, wherein the intermediate ~~node~~ connector is not aware ~~that the first node exists~~of a connection to the first connector and the first connector is not aware of a connection to the intermediate connector; and

a tuple manager to generate a new tuple identifying a relationship between the first ~~node~~ connector and the intermediate ~~node~~ connector based on the intermediate connection and the direct link, wherein the new tuple indicates that the first ~~node~~ connector is directly coupled to the intermediate ~~node~~connector.

Claim 33 (currently amended): The system of claim 32, wherein the host is a singly-heard-host which is the only host heard on the first port of the first ~~node~~connector.

Claim 34 (previously presented): The system of claim 33, wherein the singly-heard-host is at least one of a workstation, a personal computer, a terminal and a printer.

Claim 35 (previously presented): The system of claim 32, wherein the tuple and the new tuple contain data associated with a topology of the network.

Claim 36 (currently amended): The system of claim 32, wherein the tuple manager is to determine whether the first ~~node~~ connector is directly coupled to the intermediate ~~node~~ connector via a second port of the first ~~node~~connector.

Claim 37 (previously presented): The system of claim 32, further comprising:

a database to store the new tuple generated.

Claim 38 (currently amended): The system of claim 32, wherein the tuple manager is to further determine whether the host is heard only by the first port of the first ~~node~~ connector and if the host is heard only by the first port of the first ~~node~~connector, the tuple ~~manager~~ manager is to classify the new tuple as a singly-heard host link tuple.

Claim 39 (currently amended): The system of claim 38, wherein the tuple manager is to further determine if another ~~node~~ connector hears the host as a singly-heard host, classify the new tuple as a singly-heard conflict link tuple if another ~~node~~ connector hears the host and resolve a conflict associated with the host between the first ~~node~~ connector and the another ~~node~~connector.

Claim 40 (currently amended): The system of claim 32, wherein the tuple manager is to further generate an extra host link tuple for the intermediate ~~node~~ connector indirectly coupled to the host via the intermediate connection, examine the singly heard host link tuple and the extra host link tuple and generate a conn-to-conn link tuple between the first ~~node~~ connector and the intermediate ~~node~~ connector if the first ~~node~~ connector is determined to be connected to the host and the intermediate ~~node~~ connector is determined to be connected to the host.